MODCOMP

Programmer's Reference Manual
MAX III/IV

Link Loader

211-804003-B01

Link Loader

Manual History

Manual Order Number: 211-804003-B01
Title: *MAX III/IV Link Loader, Programmer's Reference Manual*
Product Number: 600599-102C.0 (MAX III) and 600404-102B.2 (MAX IV)

Revision Number	Date Issued	Description
00	07/76	Initial Issue.
01	02/78	Revision for Rev. C.0
02	06/78	Reissue
03	07/80	Reissue for Revs. C.0 and B.0
04	08/81	Reissue for Revs. C.0 and B.1
05	10/82	Reissue for Revs. C.0 and B.2

PREFACE

Audience: This manual is directed to the system programmer using the MAX III/IV Link Loader.

Subject: This manual describes the functional operation and use of the MAX III/IV Link Loader.

Product Support: The MAX III/IV Link Loader operates under the MAX III and MAX IV Operating Systems.

Related Publications: The reader is referred to the following documents for additional information. The manual order numbering system has been revised. For your convenience, the former manual order number is listed in parentheses below the new manual order number.

When ordering manuals, 'REV' is to be replaced with the revision of the manual that is listed in the publications catalog. In the actual manual number, the last three characters reflect the current product and publications revision level. For example: in the last three digits of Manual Order Number 211-804005-G00, 'G' is the current product revision level, and '00' is the current publications revision level.

If the 'REV' characters in the order do not match a current available manual revision, the latest revision will be shipped.

MANUAL ORDER NUMBER	TITLE
213-804001-REV (210-610304-000)	MAX IV GENERAL OPERATING SYSTEM System Guide
213-802001-REV (210-600303-000)	MAX III OPERATING SYSTEM System Guide
211-804002-REV (210-600500-028)	MAX III/IV NON-RESIDENT JOB CONTROL AND BATCH FACILITIES Programmer's Reference Manual
210-804001-REV (210-600500-008)	MAX III/IV ASSEMBLERS Language Reference Manual

NOTE: Those users familiar with the previous issue of this manual will notice that the format has changed significantly throughout the manual. However, revision bars are used only to reflect technical changes.

Microfiche Format: MODCOMP currently produces microfiche in two formats. Format 1 presents up to 98 frames of 8 1/2" x 11" paper. Format 2 presents up to 49 frames of 11" x 17" paper. Microfiche frames follow the same order as the pages of a paper manual, starting at frame A1.

The Table of Contents appears in the last frames of the G row of each microfiche sheet and contains a cross-reference of paper manual page numbers to the microfiche frame and sheet numbers.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1
OPERATION OF LINK LOADER

The Link Loader system processor:

- Loads a main object module, together with any additional external object modules that it references, into memory.
- Links all of the modules together to form one complete and executable set of machine code.

The program name of the Link Loader is LNKLD. On MAX III systems, the Link Loader is cataloged as an overlay to the Batch task under the name LNK. This shorter name is used from this point on. LNK can also be used by executing the MAX III system service REX,#2C. Refer to the MAX III OPERATING SYSTEM System Guide. The service is optional in all versions of the MAX III Operating System.

NOTE: The REX,#2C service and all discussions of it in this manual are applicable only to the MAX III Operating System.

In the MAX IV Operating System, the Link Loader can be invoked only through Job Control. Refer to sections 2.1 through 2.2.

1.1 LOGICAL FILES USED BY THE LINK LOADER

The following logical files are used by MAX III/IV Link Loader.

logfile Input logical file

Any user-specified logical file from which the main object module is read. The name of the logical file used for input must be specified as an argument in the REX,#2C call. Refer to figure 2-2.

LNK reads and loads the main object module and any required object modules that might follow on the specified logical file. Reading from this logical file starts from its current position, that is, no rewinding of the device occurs. If externally referenced object modules must be found, LNK subsequently reads one or both of two logical files (UL and LB) as soon as a file-mark record is encountered on the input logical file.

UL User Object Library

If the logical file UL has been assigned to a device or to another logical file that contains object modules referenced by the main object module, logical file UL is searched following the search of the input logical file. Any object modules referenced by the main object module are loaded.

LB System Library

After reading and searching the input logical file and possibly logical file UL for the externally referenced object modules, a search for still missing object modules continues on the logical file LB.

LO Listing Output

The map listing of the symbol table is written on the logical file LO.

CO Computer-to-Operator

LNK messages that are directed to the operator's attention are written on the logical file CO.

1.2 LOGIC FLOW OF THE LINK LOADER

The MAX III system service REX,#2C used with LNK does the following:

1. Loads LNK into the upper part of the memory allocated for the task under which LNK is executed.

2. Transfers control to LNK with:

   a. The address at which the main object module is to be loaded.

   b. The logical file name from which the main object module is to be read. After the main object module is read, one or both of the logical files User Object Library (UL) and System Library (LB) are searched to load object modules that were referenced by the main object module.

When the link-loading is complete, control is transferred to the start-of-execution address of the main object module. If REX,#2C + #80 is used, control is transferred to the REX call return address.

Initially, the main object module is read and loaded from the logical file specified in the REX,#2C call. A symbol table of all internals, externals, and common block names is built by LNK.

The symbol table begins at the memory location following the area where LNK was loaded and extends downwards toward the memory area

where the main object module and any additional object modules are being loaded. If the memory area allocated for the task is not large enough, the loading of an object module can cause an attempt to overlay the symbol table. In such an event, the host task is aborted.

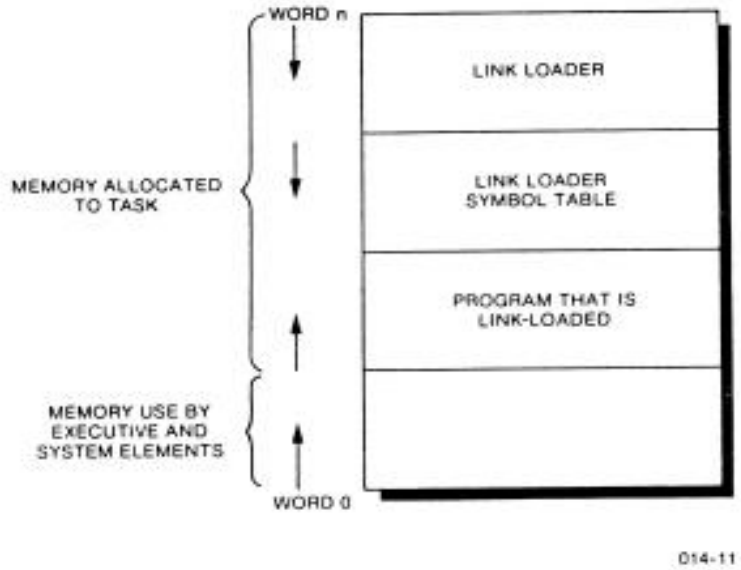

Figure 1-1. Memory Partitioning During Link-Loading

The symbol table contains information such as entry addresses for internal names and string-back addresses for unsatisfied external names. A symbolic name that is declared as an external name in an object module but is NOT used within that program is ignored by LNK and is not added to the symbol table as a missing external name.

The name of the object module to be link-loaded and the logical file from which it is to be read are specified in the MAX III REX,#2C call. LNK does NOT rewind the input file prior to the loading of a main object module. In other words, LNK begins reading the logical file used for input at the present position of that logical file.

If the name of the main object module is specified as MAIN, the first object module that is read from the input logical file is loaded together with subsequent required object modules until a file-mark record is encountered. If the program name of the main object module is specified, program names of the object modules on the logical file are checked and object modules are bypassed until the object module with the specified program name is found. This object module is loaded together with any subsequent required object modules until a file-mark record is encountered.

LNK proceeds to search the logical files UL and LB for any object modules that need to be loaded as soon as the main object module and possibly other object modules are loaded from logical file BI.

The logical file UL is searched if it is assigned to another logical file or to a device other than the NO device.

If the logical file LB is specified as the input file (for example, ASS BI=LB), LNK stops reading object modules from logical file LB immediately after loading the main object module. LNK then rewinds the logical file LB. Logical files UL and LB are then searched for externally referenced object modules, if necessary.

NOTE: The Link Loader cannot search directorized library files.

1.3 SEARCH OF THE LOGICAL FILE UL

In addition to the standard System Library (LB) logical file, the user can create and maintain a private library of object modules on the logical file UL. During link-loading, LNK first searches the logical file UL in trying to find missing externals. LNK loads and links the required object modules.

A search of the logical file LB follows the search of the logical file UL. If the logical file UL was assigned to the NO device or was not assigned at all, LNK assumes that there is no user library and proceeds to search the logical file LB.

The following two sections discuss the assignment of logical file UL to various devices.

1.3.1 UL Assignment to Paper-Tape or Card Reader

When the logical file UL is assigned to a paper-tape reader or to a card reader, LNK loads the main object module and other externally referenced object modules that follow it from the input logical file BI until a file-mark record is encountered. At that point, object modules are read and possibly loaded and linked from the paper-tape reader or card reader. If a file-mark record is encountered and if all external names are NOT satisfied, LNK reads the logical file LB. Another search of logical file UL is NOT performed.

1.3.2 UL Assignment to a Magnetic Tape or Disc Partition

When the logical file UL is assigned to a magnetic tape or a disc partition, LNK loads the main object module and other externally referenced object modules that follow it from the input logical file BI until a file-mark record is encountered. At that point, object modules are read and possibly loaded and linked from the magnetic tape or disc partition. The logical file UL is searched in the same manner

as the logical file LB. Refer to section 1.4. Another search of logical file UL can be performed because logical file UL is assigned to a rewindable device. The search of logical file UL is followed by a search of logical file LB if additional object modules are needed.

1.4 SEARCH OF THE LOGICAL FILE LB

The logical file LB is normally assigned to the disc partition that holds the System Library. LB is the last logical file searched by LNK to load and link missing object modules. The logical file LB is rewound before every search for object modules that contain external names. If all the required modules are found and loaded, the link-loading phase is completed and program execution begins.

If at least one object module was loaded during the previous search and if missing external names are still not satisfied, the logical file LB is rewound and searched again. If no object module was loaded during a search and if missing external names are still not satisfied, LNK writes the following message on the logical file CO:

```
MISSING ROUTINES
```

At this point, the host task enters the HOLD state. The operator can activate the Operator Communication (OC) task and enter:

```
/RES MAP
```

This command results in a MAP listing of the symbol table. The symbol table includes the name(s) of the missing object module(s). After making new file assignments under the OC task, the operator can enter

```
/RES
```

This command continues the search for missing object modules on logical file LB. An example of a MAP listing is shown in figure 2-1.

If logical file LB is assigned to a paper-tape reader or a card reader, the message REWIND LIBRARY is written to the logical file CO at the beginning of every search of logical file LB. After this message is displayed, the host task enters the HOLD state to allow the operator to prepare the paper-tape or card library. When the library is ready, the user activates the OC task and continues the execution of LNK by entering:

```
/RESUME
```

1.5 DATA STRUCTURE

The Link Loader reads records in MODCOMP's Standard Binary Object Format as described in the Macro Assembler binary output format in the MAX III/IV ASSEMBLERS Language Reference Manual.

CHAPTER 2
GUIDE TO THE USE OF LINK LOADER

2.1 LINK LOADER USE BY JOB CONTROL

In the MAX III environment, the arguments for the REX,#2C Link Loader service can be passed to the operating system using the $EXECUTE directive in Job Control. In addition, the arguments can be directly passed from an assembly-language program through a REX call (refer to section 2.3).

In the MAX IV environment, the only manner in which LNK is executed is as a Job Control overlay through the $EXECUTE Directive. Refer to the MAX III/IV NON-RESIDENT JOB CONTROL AND BATCH FACILITIES Programmer's Reference Manual, specifically to the $EXECUTE Directive in chapter 3 and to section 4.4.2.

The Link Loader loads into memory object modules of the format produced by a MAX III assembler or compiler. When the Link Loader is used on a MAX IV system, it cannot take full account of counters and attributes (caused by the CTR and ATR directives to the Macro Assemblers and present in some MAX IV Language Processor output) or of global or extended common. The Link Loader allows a counter (without attributes) to be declared at the start of a module. Later declaration of a different counter or declaration of any attributes causes an illegal function code error.

The Link Loader can only access libraries in sequential format. It cannot access object libraries in directory format.

2.1.1 Syntax of $EXECUTE Directive

```
                1              2          3          4          5
$EXE     MAIN,           logfile,    LINK  [,NOMAP]  [,HOLD]
         program-name,
```

MAIN program-name	Parameter 1 (required) specifies one of the following: MAIN specifies that the first object module encountered on the input logical file is the main object module in the link-loading process. program-name specifies the 1- to 6-character program name of the main object module to be link-loaded and executed.

logfile — Parameter 2 (required) specifies the name of the logical file from which the main object module is to be loaded. The reading and loading of object modules from the logical file starts at the current position of the logical file. No rewinding occurs.

If a file-mark record is encountered before the main object module is found, the host task aborts with the abort code EF. Refer to appendix A.

LINK — Parameter 3 (required) specifies to Job Control that LNK is to be used for link-loading and executing the main object module.

[,NOMAP] — Parameter 4 (optional) specifies that the MAP listing of the symbol table should NOT be produced.

[,HOLD] — Parameter 5 (optional) specifies that the task under which LNK is executed enters the HOLD state at the start of the execution of LNK.

NOTE: If a user library is to be used for link-loading the program, the logical file UL must be assigned to the device or logical file that holds the library before initiating the Link Loader.

2.1.2 Example of Link Loader Use by Job Control

Logical files in the following example default as follows:

```
CI=CR      Card Reader
SI=CR      Card Reader
BO=DSA     Scratch Disc Partition A
SO=DSB     Scratch Disc Partition B
SC=DSM     Scratch Disc Partition M
LO=TYC     Listing Device
CO=TY      Console Keyboard/Printer
LMB=DBB    Load Module Disc File
```

An example of a job stream to compile and execute a program named PROG1 and its referenced subroutine SUBR1 follows. In this job stream, the object modules of both the main program and the subroutine reside on the same logical file, that is, BO.

```
$JOB
$ASS  SO=SC
$EXE  FRX
C     MAIN PROGRAM
      PROGRAM PROG1
      DIMENSION K(3)
      DATA K/1,2,3/
      I=K(1)
      CALL SUBR1 (K)
      END
```

MAIN PROGRAM SOURCE DECK

```
C       SUBROUTINE SUBR1
        SUBROUTINE SUBR1 (N)
        DIMENSION N(3)
        N(3) = N(1)+1
        RETURN
        END
```

SUBROUTINE SOURCE DECK

```
$$
$WEO SO
$ASS SI=SC
$EXE M2A,,NOSC,NOLO
$WEO BO
$REW BO
$EXE MAIN,BO,LINK,NOMAP
```

FILE MARK

The following job stream may be used when the object modules of the main program and the subroutine reside on different logical files.

```
$JOB
$ASS SO=SC
$EXE FRX
...
...
...
$$
$WEOF SO
$REW BO
$ASS SI=SC
$EXE M2A,,NOSC,NOLO
$WEOF BO
$ASS SO=SC SI=CR
$EXE FRX
...
...
...
$$
$ASS SI=SC SO=SO BO=SO
$EXE M2A,,NOSC,NOLO
$WEOF BO
$ASS UL=SO BO=BO
$REW BO
$EXE MAIN,BO,LINK
```

MAIN PROGRAM SOURCE DECK

FILE MARK

SUBROUTINE SOURCE DECK

FILE MARK

In the preceding job stream, logical file UL is assigned to SO because the object code of the user subroutine SUBR1 resides on the disc partition (DSB) to which logical file SO is assigned. Other required subroutines reside on the System Library logical file (LB).

In addition, MAIN is used as the program name in the $EXE MAIN,BO,LINK Directive because logical file BO was rewound, thereby making PROG1 the first object module read from logical file BO.

The program name PROG1 could have been used instead of MAIN (for example, $EXE PROG1,BO,LINK) and the directive would have functioned identically.

Before the execution of PROG1, a map of the symbol table is listed on the logical file LO. The map listing associated with the example follows.

```
*C LINK LOADER MEMORY MAP     DATE 12/27/82
P     PROG1       B000       B080
C     LBF:        B000
      BRX::       B091
P     SUBR1       B0C8
      SUBR1       B0CA
P     FPA:        B0D6
      FPA:        B0D6
P     EXIT        B0FC
      EXIT        B0FC
P     L:ERR       B0FD
      L:ERR       B0FD
P     FRT:5       B14D
      BRX:        B156
      TP2C:       B14E
      TP2B:       B14D
*HIGH      B156
*XFER      B0B6
```

Figure 2-1. Example Map Listing

The 1-character label in front of a name in the map specifies the nature of that name. The characters used are:

C - COMMON NAME
P - PROGRAM NAME
U - UNDEFINED or MISSING NAME

*HIGH denotes the last memory location used.

*XFER denotes the transfer address.

Names not preceded by a character label are internal names (entry points) in the object module listed directly above the internal name.

All addresses in the map listing are hexadecimal addresses. A second value after the program name shows the location at which coded instructions start. In the previous example, a 128-word area is saved at the beginning of the program as a FORTRAN buffer area.

2.1.3 Example of a Procedure

The following Job Control procedure may be used to link-load and execute a program as an overlay of Job Control.

```
$PROCEDURE LINKX
$ASS BI=%1 UL=NO
$IFM %3,2
$ASS UL=%3
$REW UL
$IFM %4,P=OPTION NOMAP
$IFP %5,P=OPTION HOLD
$IFP %2,P=$EXE %2,BI,LINK;$EXE MAIN,BI,LINK
$NOTE END-OF-EXECUTION OF %2
```

The above procedure is invoked by a statement in the following form:

```
                1           2        3     4     5
$[DO ]LINKX,logfile[,program-name][,ul][,map][,hold]
```

logfile — Parameter 1 (required) specifies the name of the logical file from which the main object module is to be read.

pname — Parameter 2 (optional) specifies a 1- to 6-character program name of the main object module to be link-loaded and executed. If not specified, the first object module that is read from the input logical file is taken as the main object module and is link-loaded and executed.

ul — Parameter 3 (optional) specifies the name of the logical file containing the User Object Library from which subprograms required by the main object module are read. If not specified, the logical file UL is NOT searched during the link-loading because it is assigned to the NO device.

map — Parameter 4 (optional) specifies the need for a map listing. If any character(s) are entered for this parameter, the map listing of the symbol table is produced.

hold — Parameter 5 (optional) specifies the need to enter the HOLD state. If any character(s) are entered for this parameter, the host task enters the HOLD state at the start of the Link Loader execution.

2.2 OPTIONS

HOLD — If the HOLD option is set, the host task enters the HOLD state at the start of the Link Loader execution. To proceed with the link-loading, the Operator Communication task must be activated and a /RESUME Directive must be entered.

MAP If the MAP option is set, the map listing of the symbol table is produced.

2.3 ASSEMBLY LANGUAGE REX CALL FOR THE LINK LOADER

If the RETURN option is NOT used in the REX call, the program is executed after the link-loading is complete. Refer to the MAX III OPERATING SYSTEM System Guide for information on the REX, #2C system service.

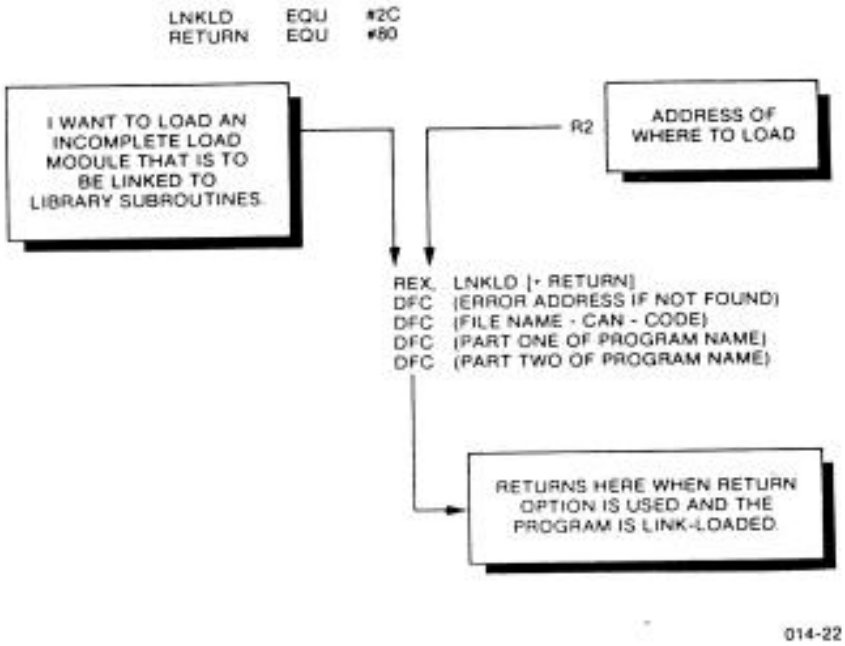

Figure 2-2. Assembly Language REX Call for the Link Loader

2.4 LIMITATIONS

During link-loading, LNK places string-back addresses at memory locations where addresses of external labels are to be stored when object modules containing those labels (internal labels) are found and loaded.

A loading problem arises if:

- An external label is referenced at some location within an object module being loaded by LNK.
- And a subsequent loading operation changes the content of that same location (that is, the location reserved for the string-back address).

The following two programs provide an example.

```
(1)         PGM PROG1
      AREA  COM #50
      A     CEQ AREA
            INC AREA+4
            DFC XT
            ...
            ORG 0
            EXT XT
            ...
            END


(2)         PGM PROG2
      AREA  COM #20
            ...
            INC AREA+4
            DFC LABEL
            ...
            ORG 0
            ...
     LABEL  DFC VALUE
            END
```

After LNK loads PROG1 into memory, the memory location AREA+4 contains a string-back address that is used by LNK when the object module containing the external label XT is found and loaded. However, if LNK loads program PROG2 next, the content of the same memory location (AREA+4) is altered and contains the address of LABEL. Thus, the string-back address previously stored at that memory location is destroyed.

The user must avoid referencing external labels from within common areas in programs. In addition, use of the Assembler Directive ORG in a way that causes external references to be overlaid must be avoided.

APPENDIX A
ERROR MESSAGES

When a link-loading error is encountered, the host task is aborted (REX,#13) with a 2-character abort code indicating the type of error. The format of the error message is:

```
!ABORT (xxnnnn)
```

The variable xx represents the abort code and the variable nnnn represents the address. The error messages are output to logical file CO.

Example:

```
!ABORT (HIB156)
```

System abort conditions encountered while reading binary object records are flagged with the following abort codes:

FC - Illegal function code
XT - Error in an external name reference
CM - Error in a common definition
SQ - Sequence error
CK - Checksum error

Link Loader abort codes are:

EF A file-mark record was read on the logical file from which the main object module was to be read before the main object module under the program name specified was found.

HI An object module that was being loaded was about to overlay the LNK symbol table.

Link Loader error and informational messages follow. These messages are displayed on logical file CO.

MISSING ROUTINES	No object module was loaded after a search of logical file LB and unsatisfied external names still exist.
REWIND LIBRARY	In case logical file LB is assigned to a paper-tape or card reader, LNK notifies the operator that the paper-tape must be rewound or the card deck prepared for another search of logical file LB.

INDEX

Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 3624 FT. LAUDERDALE, FL 33309

POSTAGE WILL BE PAID BY ADDRESSEE

MODULAR COMPUTER SYSTEMS
1650 W. McNAB ROAD
P.O. BOX 6099
FT. LAUDERDALE, FLORIDA 33310

Attention: TECHNICAL PUBLICATIONS, M.S. #85

Fold

MODCOMP

Please comment on the publication's completeness, accuracy, and readability. We also appreciate any general suggestions you may have to improve this publication.

If you found errors in this publication, please specify the page number or include a copy of the page with your remarks.

Your comments will be promptly investigated and appropriate action will be taken. If you require a written answer, please check the box and include your address below.

☐

Comments: _____

Manual Title _____

Manual Order Number _____ Issue Date _____

Name _____ Position _____

Company _____

Address _____

_____ Telephone ( ) _____

MCS 148D (1/83)

MODCOMP

Corporate Headquarters
MODULAR COMPUTER SYSTEMS, Inc., 1650 West McNab Road, P.O. Box 6099, Ft. Lauderdale, FL 33310 Tel (305)974-1380, TWX 510-956-9414

European Headquarters
MODULAR COMPUTER SERVICES, INC., Eskdale Road, Winnersh, Wokingham, Berkshire, RG11 5TR, United Kingdom Tel: 0734-696888, TLX: 849149

SALES & SERVICE LOCATIONS THROUGHOUT THE WORLD

Printed in U.S.A.